INPUT-cve\_single\_technique.xlsx

Techniques.xlsx

How do we make these files?

A public dataset is available on Github-(<https://github.com/center-for-threat-informed-defense/attack_to_cve/blob/master/Att%26ckToCveMappings.csv>). This dataset has mapping of 840 cves to numerous attack techniques.

Our next task is to make this file such that we get a single cve to technique map on each row along with the cve description.

First we make the Final\_AttackToCveMappings.xlsx file.

How do we do so?

Follow code.ipynb - This script processes and merges two datasets to produce a clean, final Excel file combining CVE information with attack technique mappings. It begins by loading the file Att&ckToCveMappings.csv, and merges four columns — 'Primary Impact', 'Secondary Impact', 'Exploitation Technique', and 'Uncategorized' — into a new single column called 'Merged\_Impact\_and\_Method', while handling any missing columns by filling them with empty strings. The merged result is saved as Updated\_Att&ckToCveMappings.csv. Then, the script removes five columns: 'Primary Impact', 'Secondary Impact', 'Exploitation Technique', 'Uncategorized', and 'Phase' from this updated file, and saves the cleaned result to Final\_AttackToCveMappings.csv. Finally, the script merges Final\_AttackToCveMappings.csv (which has 'CVE\_ID' and 'Merged\_Impact\_and\_Method') with CleanedUpdates2.csv (which has 'CVE\_ID' and 'CVE\_Description') using 'CVE\_ID' as the key. The final combined dataset containing 'CVE\_ID', 'Merged\_Impact\_and\_Method', and 'CVE\_Description' is saved as an Excel file named Final\_AttackToCveMappings.xlsx.

Now our task is to convert this file into cve\_single\_techniques.xlsx :

Refer to code2:

This Python code uses the pandas library to process a CSV file named **"Final\_AttackToCveMappings.csv"** that contains mappings between CVEs and MITRE ATT&CK techniques. It first loads the CSV file into a DataFrame. Then, it renames the column **'Merged\_Impact\_and\_Method'** to **'MITRE\_Technique\_Numbers'** to standardize the column name for further processing. The code assumes that the **'MITRE\_Technique\_Numbers'** column contains multiple technique IDs separated by semicolons and spaces ("; "), so it splits these strings into lists of individual techniques. Using the **explode()** function, it expands the DataFrame so that each technique from these lists gets its own separate row, effectively creating one row per CVE-technique pair. Finally, it saves this transformed DataFrame into a new Excel file named **"cve\_single\_technique.xlsx"**, without including the DataFrame index in the file. The printed message confirms the successful creation of this output file with one technique per row.

How did we get cleanupdates 2?

This Python script automates the process of downloading, processing, and cleaning CVE (Common Vulnerabilities and Exposures) data from the National Vulnerability Database (NVD) via their REST API. It begins by creating a directory named data/ and manages two helper files: data/last\_index.txt to remember the last fetched index for resuming interrupted downloads, and data/failed\_batches.txt to log any failed API requests for retrying later. The script queries the NVD API (https://services.nvd.nist.gov/rest/json/cves/2.0) in batches of 1000 results, using a retry mechanism with exponential backoff to handle intermittent failures. All successfully fetched CVE records are appended to a list and then saved as a raw JSON file at data/all\_cves.json. In the next phase, this JSON file is loaded, and for each CVE entry, only the CVE ID and its English Description are extracted—ignoring additional metadata such as CVSS scores or CWE IDs. The cleaned data is then converted into a pandas DataFrame containing just these two fields. Finally, the cleaned output is saved as an Excel file named CleanedUpdates2.xlsx in the ../Data/ directory. This results in a simplified and readable dataset ready for further analysis or mapping.

How did we get techniques?

Code Accessig df-This Python script uses the mitreattack-python library to retrieve and process data from the MITRE ATT&CK framework. It begins by downloading the full STIX representation of the **Enterprise ATT&CK dataset** using the attackToExcel.get\_stix\_data() function. The raw STIX data is then parsed into structured pandas DataFrames using stixToDf.techniquesToDf(), which returns information about techniques and their related metadata. From the resulting data, only the **techniques DataFrame** is selected, and two specific columns—ID (such as T1102) and description (a textual explanation of each technique)—are retained for further use. The script then saves this cleaned subset of the data to an Excel file named techniques.xlsx in the working directory. This output provides a concise and readable reference of all Enterprise ATT&CK techniques and their descriptions, ideal for analysis, reporting, or downstream processing.